Core Image

Session 202

These are confidential sessions—please refrain from filming or taking pictures.

Graphics and Media

Core Image

Ralph Brunner
Senior Vice Mineralogist

http://developer.apple.com/wwdc2005/

Hidden Agenda

- What Core Image is all about
- A quick dash through the sample code underbrush
- CIAnnotation—the star app of the day
- Two Image Units to go
- Poetry in Motion™
- Pointers to Arrays of Sessions

Meet Core Image

- New image processing framework in Mac OS X Tiger
- Enables GPU accelerated image processing
- Full floating-point, color-managed pipeline
- Base set of 100 filters
- Image Units to extend the base set
- Core Image filters can execute on CPU if adequate GPU is not available
- Compiles to Velocity Engine and SSE3

Pimp Your App With Some Bling From Core Image

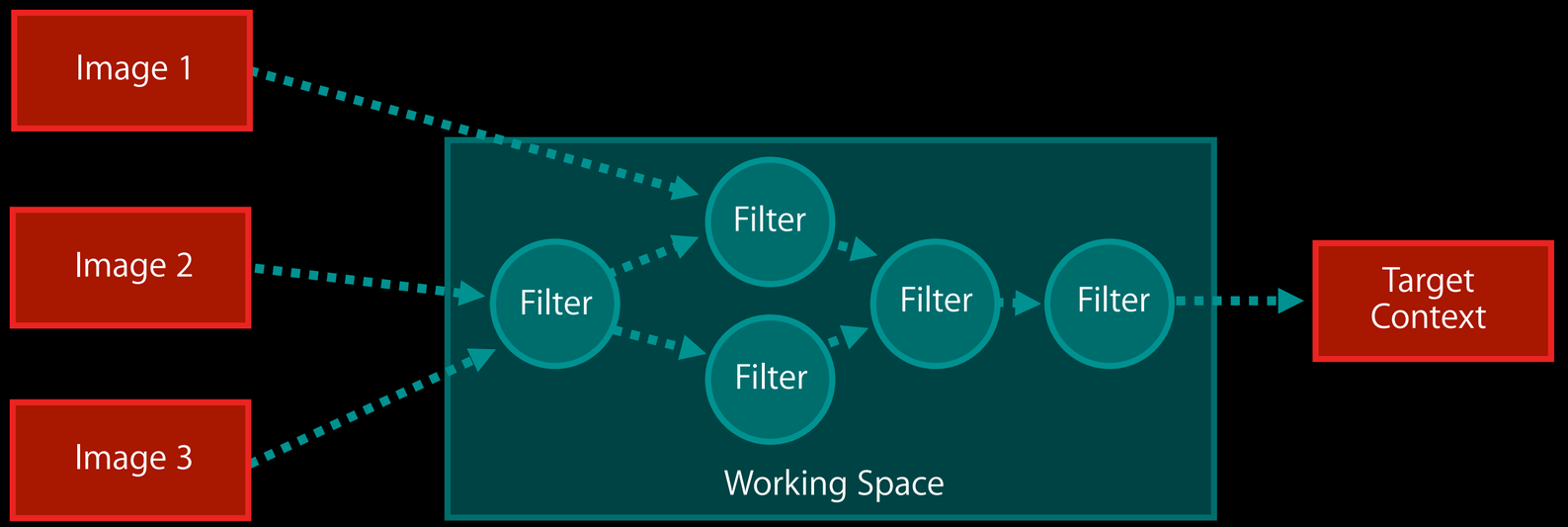
- Allow in-place image editing
- This millennium, image processing is done in the display pipeline
- Add color controls to video
- Host Image Units if appropriate
  - Allow other developers to extend your app
- Write custom filters as Image Unit, so that they can be used in other apps
- Use effects to enhance UI (apply some taste…)

About the Man Behind the Curtain

- Core Image is fundamentally lazy
- Getting an output image from a filter does not actually render it
- Rendering is deferred until draw time
- Deferred evaluation allows concatenation of filter operations
- There is a just-in-time compiler under the hood to facilitate that
- Computons to Blittons ratio is 10:1 for CPU, 4:1 for GPU

Random Acts of Color Management

- Color matching on input and output
- Default working space:“Generic RGB HDR”
- Generic RGB HDR has important properties for image processing
  - Light linear
  - Infinite gamut



Demo

Code Commences Now

http://developer.apple.com/wwdc2005/

A Journey Into the CIAnnotation App

Frank Doepke
Low Orbit Intelligence Analyst

http://developer.apple.com/wwdc2005/

One Slide Before…

- CIAnnotation sample app
  - Dealing with large, detailed satellite imagery
  - No comment here but I have a red marker
  - Do not touch the original

The Ten Meter Resolution View of the App

- Single document Cocoa app
  - AnnotationController: the application controller
  - AnnotationDocument: your guess…
  - AnnotationView: view subclass for mouse handling
- Layers—just like onions
  - CILayer: abstract base class
  - CITextLayer: text objects and rendering of the text
  - PaintLayer: uses a CIAccumulator with the Paint Image Unit
  - CIImageLayer: handles the background image we want to annotate
- Utilities
  - SampleCIScrollableView: Modified SampleCIView that allows scrolling
  - Utilities: bag of tricks
  - OverlayWindow: Window subclasses for the text editing and the palette

The Five Meter Resolution View of the App

- Viewers like you—what makes the view scroll
- Getting text editing on top of OpenGL done
- Import and Export without Customs
- Compositing in CI

Viewers Like You—What Makes the View Scroll

- SampleCIView
  - SampleCIView has been used in our Sample Code so far
  - A simple subclass of NSOpenGLView that does the OpenGL work for you
  - Supports selective drawing/partial updates
- SampleCIScrollableView
  - Setup the orthographic projection matrix to the visible rect of the enclosing NSScrollView—this transforms the OpenGL coordinate system
  - Still uses the same interface as SampleCIView
  - Has all the same bells and whistles

Compositing in CI

- CISourceOverCompositing is your friend
  - Better than just drawing the images on top of each other
- Transitions do compositing too
  - Adding some spice to it

The One Meter Resolution View of the App

- Let’s look at the layers
  - We dive into the code: header first—no cannonballs please
  - You can paint too
    - Using the CIAccumulator
  - Your image is important—let’s scale it down
- The document holds things together
  - What pieces are involved
  - Creating the resulting image

Getting to the Bottom of the Image Units

Mark Zimmer
Expert Pixel Enchanter

http://developer.apple.com/wwdc2005/

The Paint Image Unit

- MicroPaint example shows CIImageAccumulator

The Paint Image Unit

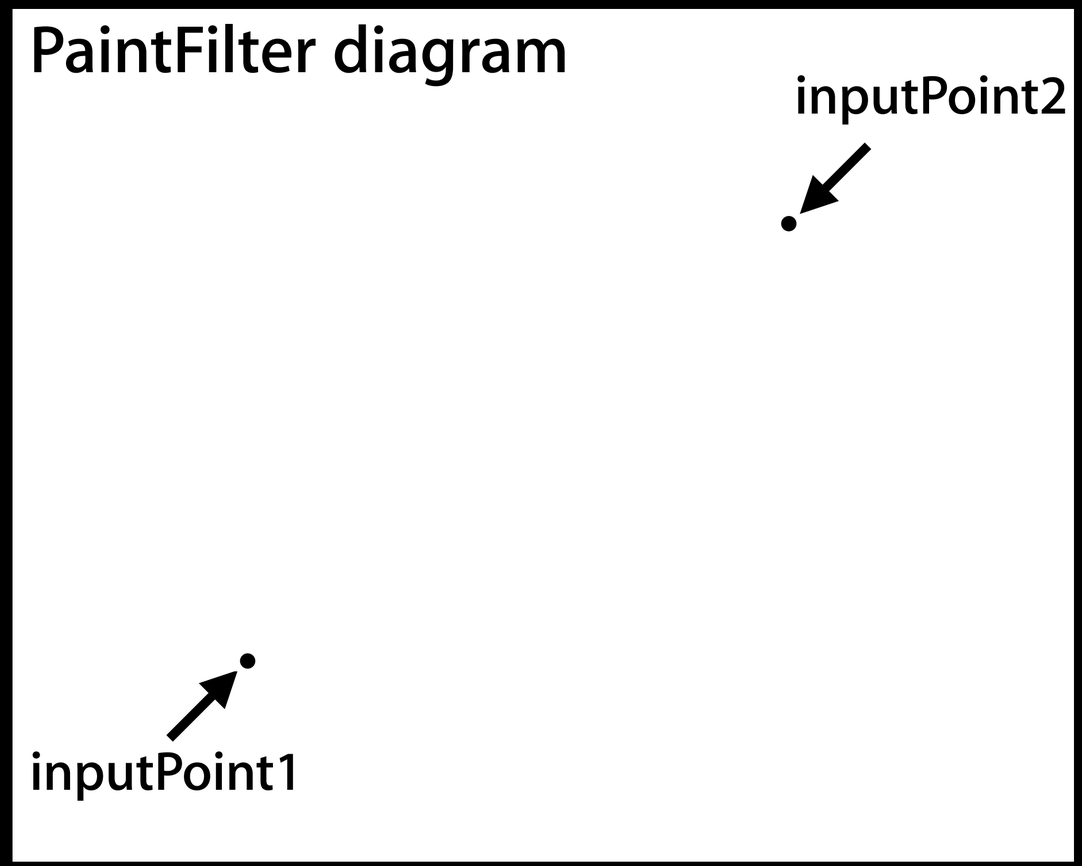
- MicroPaint example shows CIImageAccumulator
- Problem: paint strokes are not always continuous!

The Paint Image Unit

- MicroPaint example shows CIImageAccumulator
- Problem: paint strokes are not always continuous!
- Solution: render a lozenge instead of a circle

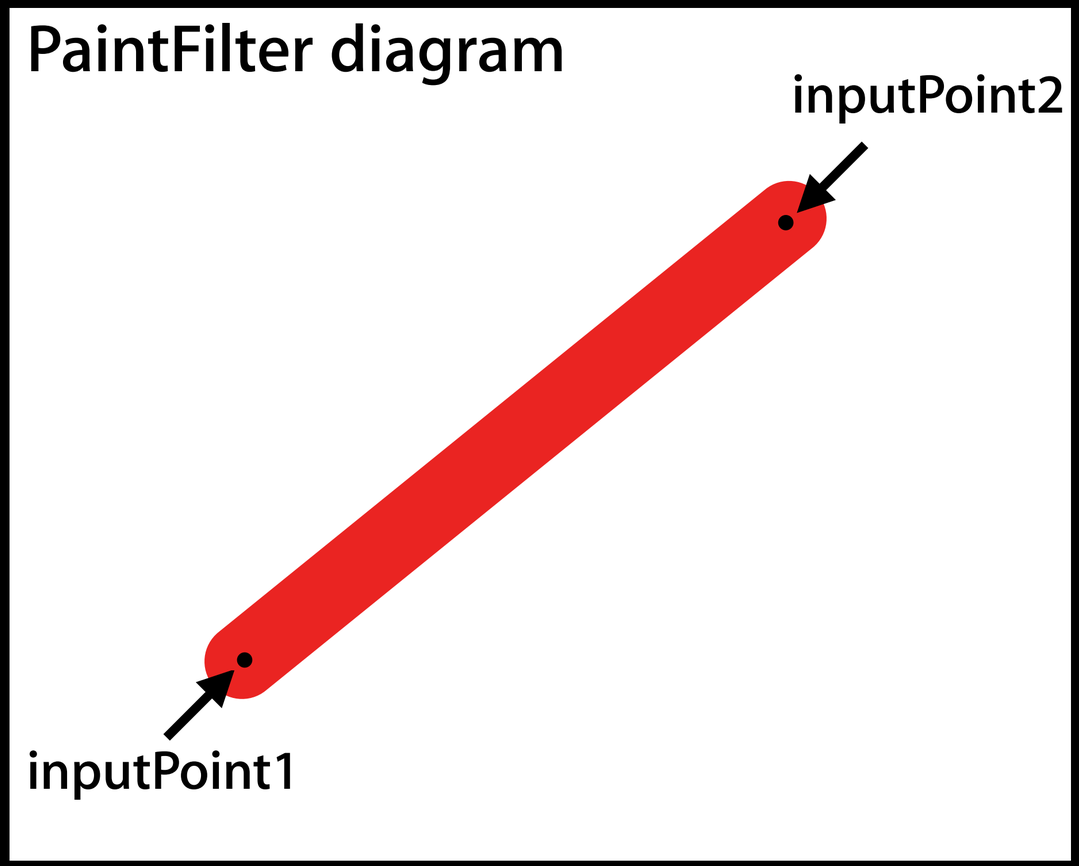
Paint Image Unit Details

Render from inputPoint1 to inputPoint2



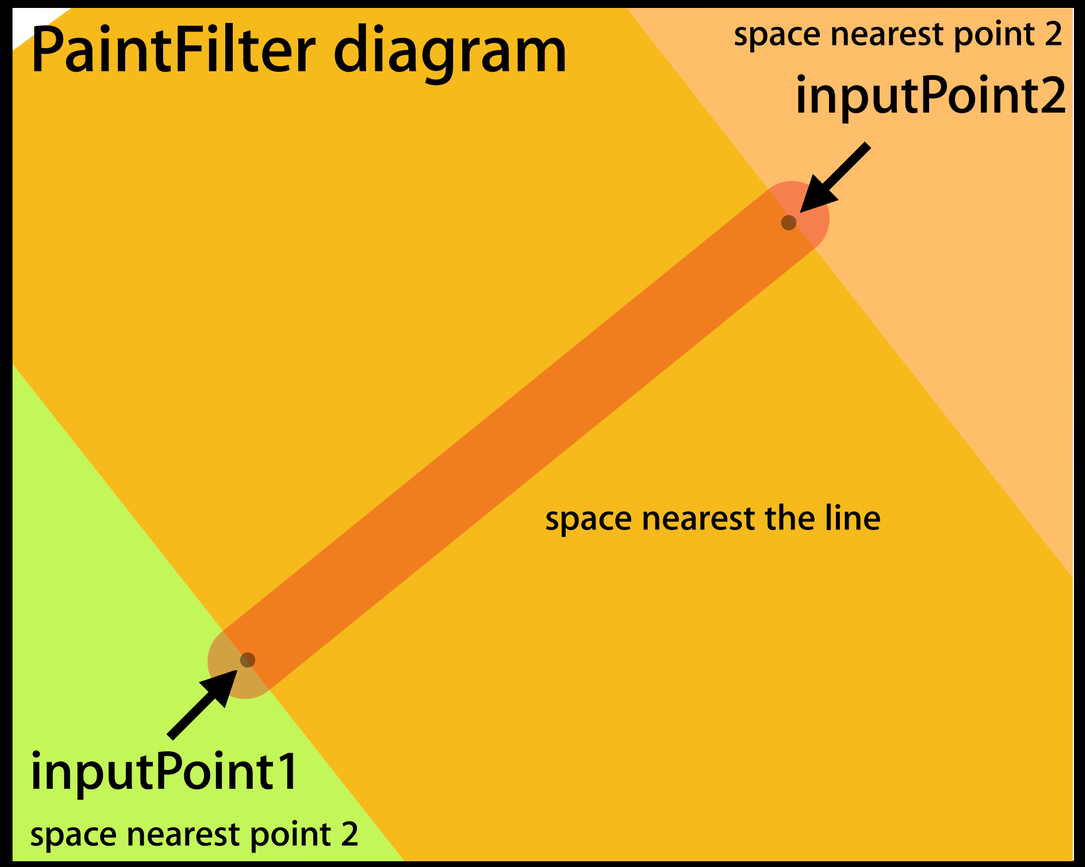
Paint Image Unit Details

With width inputWidth, using circular end caps



Paint Image Unit Details

Partition the rendering using 2D halfspaces



- Evaluate composite distance function
- 2D dot product to get halfspaces
- Euclidean distance to make circles
- 2D cross product for distance from line
- Threshold distance to make lozenge alpha
- Fill with inputColor

The Lens Image Unit

- Requirements
  - It must magnify without blurring
  - Embedded in a ring to give it weight
  - Needs a shine on it!
- Ideas for extension (fun stuff you can do)
  - Chromatic aberration
  - Some distortion inside the magnifier?
  - Caustic under the lens!

Lens Image Unit Details

InputImage is a parameter

Lens Image Unit Details

Magnify inside an anti-aliased circle using inputMagnification

- We use an additional CISampler with its own affine transform for the magnified data
- The transform is adjusted to counteract the view transform and get to individual pixels

Lens Image Unit Details

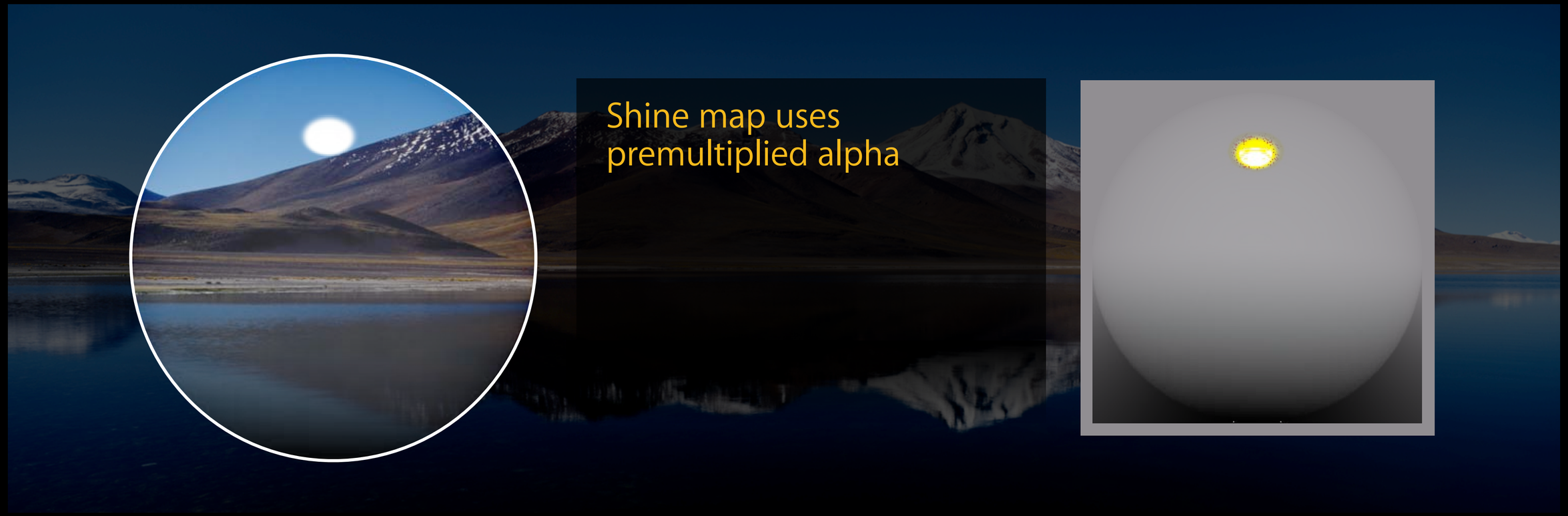
Magnify inside an anti-aliased circle using inputMagnification

Lens Image Unit Details

Circle is defined by inputCenter and inputWidth

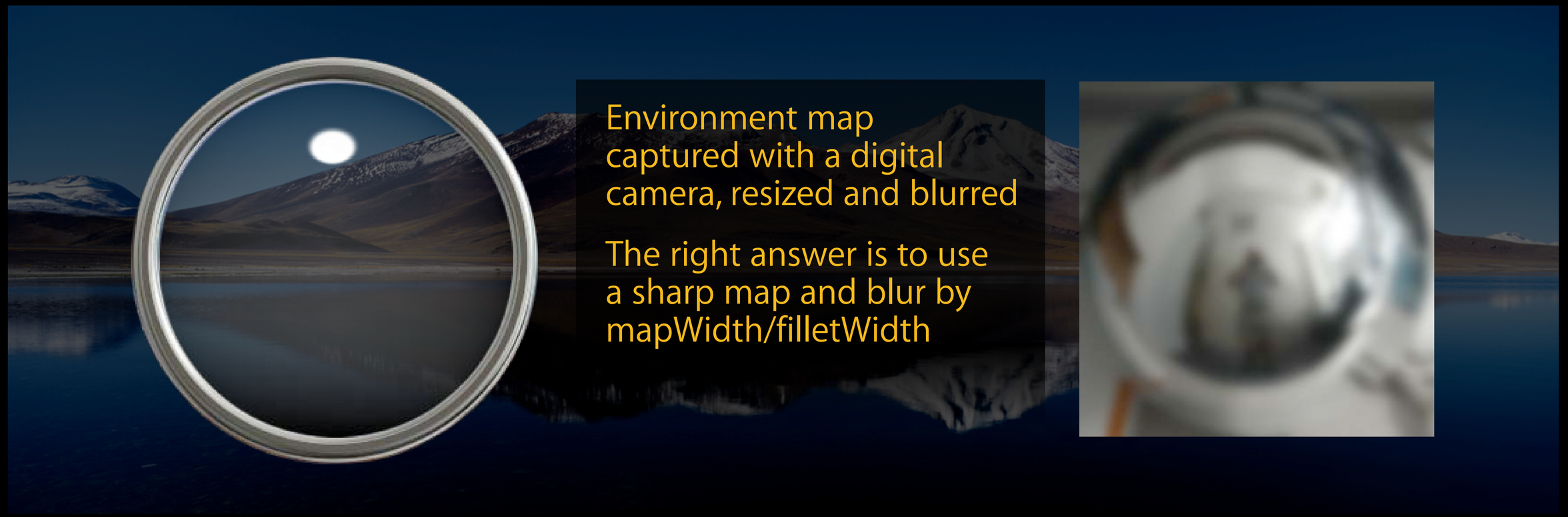
Lens Image Unit Details

Overlay a shine by inputShineOpacity, using inputRoundness



Lens Image Unit Details

Place a ring around the lens, fill using an environment map

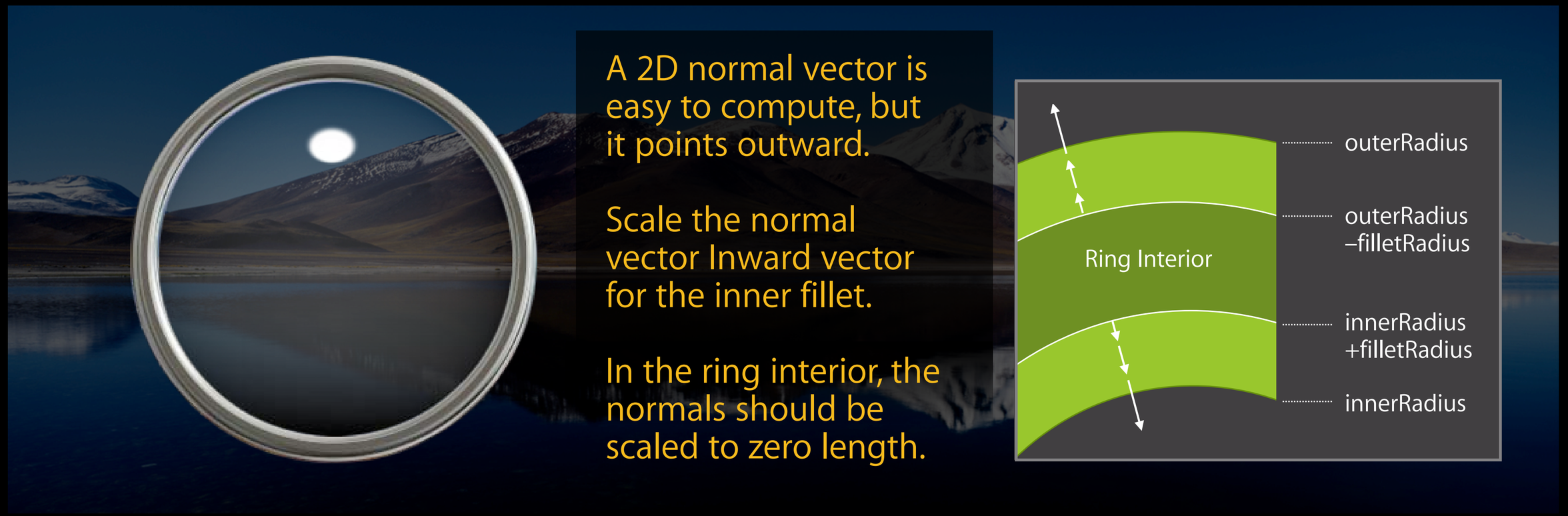


Environment map captured with a digital camera, resized and blurred

The right answer is to use a sharp map and blur by mapWidth/filletWidth

Lens Image Unit Details

Place a ring around the lens, fill using an environment map



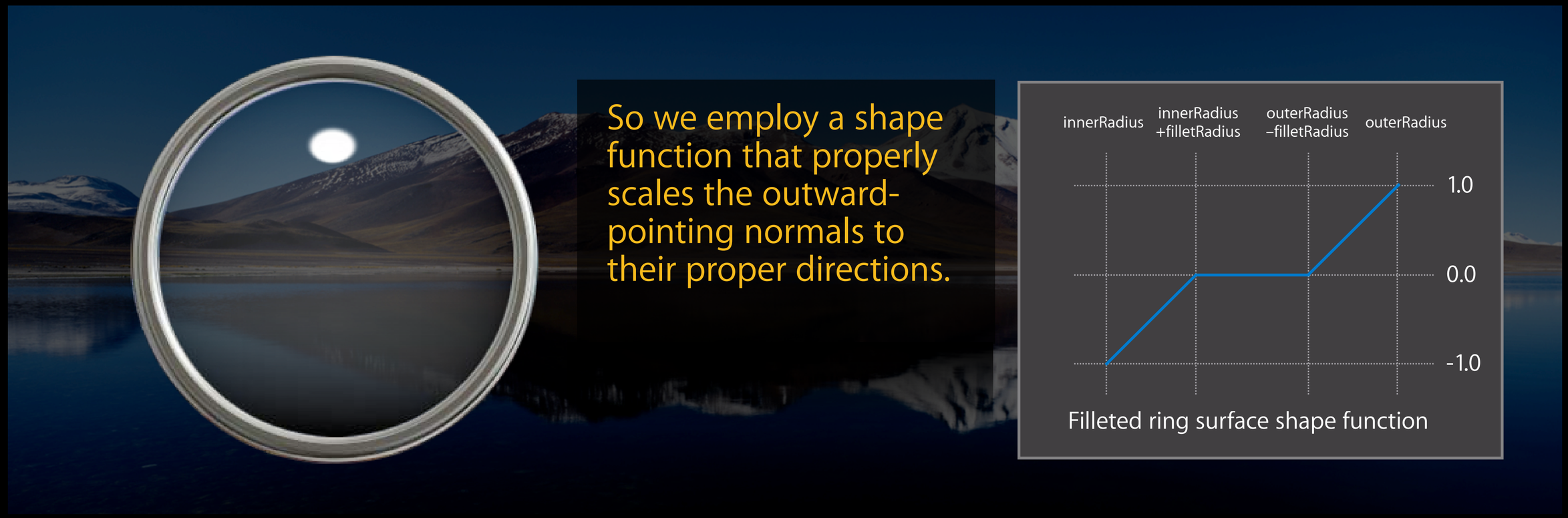
A 2D normal vector is easy to compute, but it points outward.

Scale the normal vector Inward vector for the inner fillet.

In the ring interior, the normals should be scaled to zero length.

Lens Image Unit Details

Place a ring around the lens, fill using an environment map



So we employ a shape function that properly scales the outward-pointing normals to their proper directions.

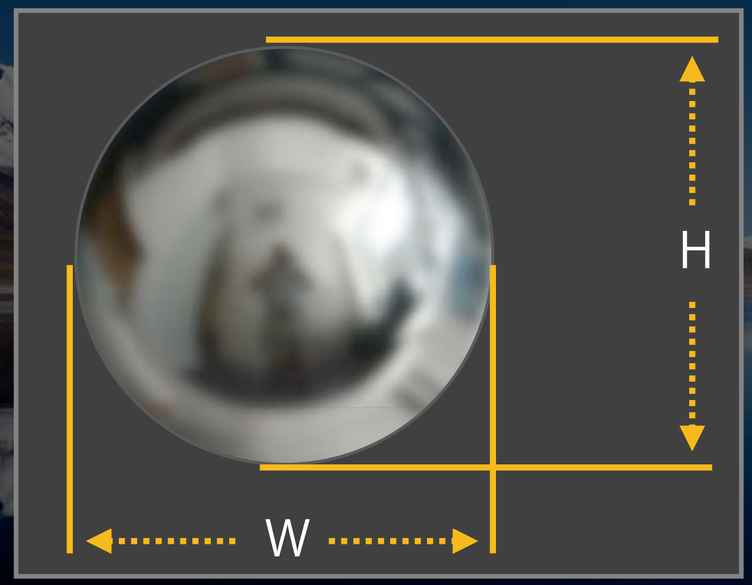
Filleted ring surface shape function

Lens Image Unit Details

Place a ring around the lens, fill using an environment map

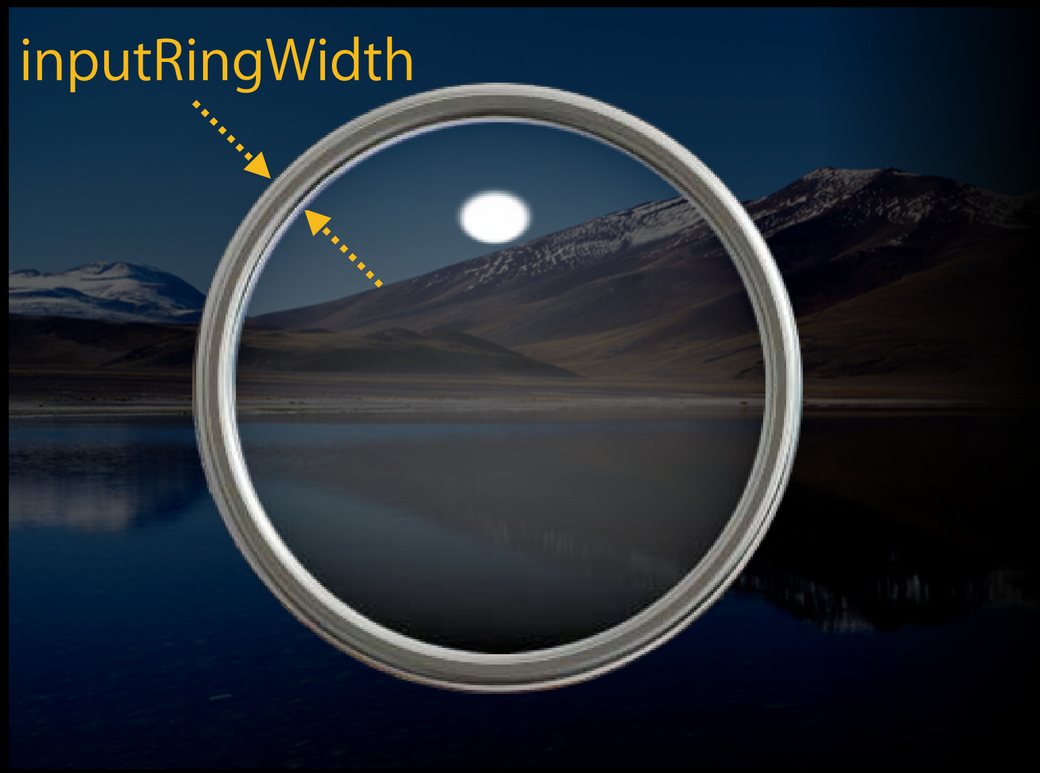
2D normal vectors always range from [-1...+1, -1...+1].

They must be scaled onto [0...w, 0...h] for use with our environment map.



Lens Image Unit Details

Magnify inside an anti-aliased circle using inputMagnification



- Fillet radius is typically half the ring width
- Edges of ring must be anti-aliased

Don’t Squeeze the Image Units!

Ralph Brunner
Having The Last Word

http://developer.apple.com/wwdc2005/

Join the Program!

- Create Image Units
- Build Image Unit hosts
- Logo program

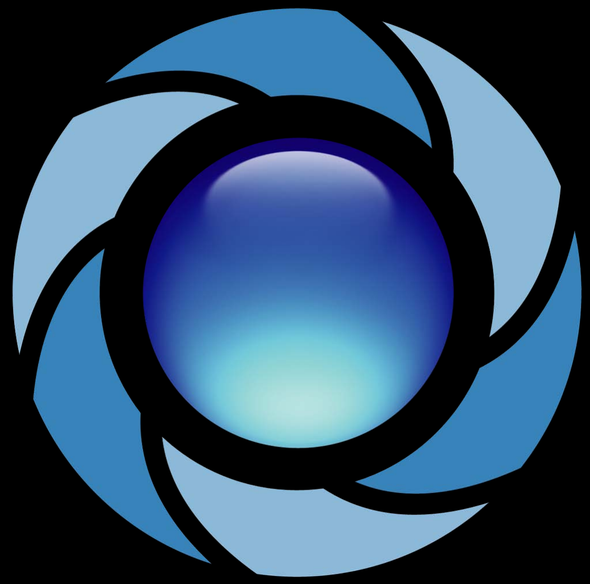


Image Units

Perpetual Motion Devices…

Fun demo commences now

http://developer.apple.com/wwdc2005/

Related Sessions

Developing Image Units	Tiger Lab Tuesday 12:00 noon
For Core Video / QuickTime integration: High-Performance Video in Mac OS X Tiger	Session 207 Wednesday 10:30AM
For Image Units in Motion: FxPlug Overview	Session 305 Thursday 3:30PM
For UI effects and AppKit integration: Cocoa Advanced View Techniques	Session 146 Friday 3:30PM
For Image Units in Motion: FxPlug in Depth	Session 308 Friday 10:30AM

Who to Contact

Travis Brown
Graphics and Imaging Evangelist
travis@apple.com

Q&A

Travis Brown
Graphics and Imaging Evangelist
travis@apple.com

http://developer.apple.com/wwdc2005/

WWDC2005

Worldwide Developers Conference